ESB solutions
White paper

WebSphere® software

Increasing IT flexibility with IBM WebSphere ESB software.

By Beth Hutchison, Marc-Thomas Schmidt and Chris Vavra, IBM Software Group

November 2006

Contents

Introduction

To encourage business growth while still keeping costs in check, many companies are searching for ways to increase the flexibility and reuse of their existing IT assets. Service oriented architecture (SOA) offers a means to define services that represent a repeatable task, such as "check customer account," using well-defined, standards-based interfaces. An enterprise service bus (ESB) provides a connectivity infrastructure that you can use to integrate services within an SOA. Together, SOA and an ESB help to reduce the number and complexity of interfaces, enabling you to focus on your core business issues, rather than on maintaining your IT infrastructure.

This white paper describes how you can take advantage of the benefits offered by an ESB using IBM WebSphere® Enterprise Service Bus (WebSphere ESB) software. With this leading-edge product, you can develop and deploy an ESB that enables you to add new services faster and change services more easily, while reusing your existing assets.

SOA and the ESB

SOA offers a flexible, extensible and composable approach to reusing and extending existing applications and services, as well as constructing new ones. You can implement services using a wide range of technologies, including IBM CICS®, IBM IMS™, Java™ 2 Platform, Enterprise Edition (J2EE), Enterprise JavaBeans (EJB), Java classes, IBM DB2® queries or Microsoft® .NET. In SOA interactions, service providers advertise the capabilities they offer by declaring the interfaces that they implement and their policies governing potential partner interactions. Service requesters can also declare the interfaces they require and the partner interactions they support. Web Services Description Language (WSDL) and other Web services standards provide the vocabulary for these declarations. Service requesters send requests to service providers that offer the capabilities they require, unaware of their implementations. SOA therefore provides an ability to virtualize business functions by isolating service definition and usage from the underlying service implementation.

An ESB helps you maximize the flexibility of an SOA. Participants in a service interaction are connected to the ESB, rather than directly to one another. When the service requester connects to the ESB, the ESB takes responsibility for delivering its requests, using messages, to a service provider offering the required function and quality of service. The ESB facilitates requester-provider addresses despite mismatched protocols, interaction patterns or service capabilities. An ESB can also enable or enhance monitoring and management. The ESB provides virtualization and management features that implement and extend the core capabilities of SOA. The ESB virtualizes:

- Location and identity. *Participants need not know the location or identity of other participants. For example, requesters need not be aware that a request could be serviced by any of several providers; service providers can be added or removed without disruption.*
- Interaction protocol. *Participants need not share the same communication protocol or interaction style. A request expressed as SOAP over HTTP can be serviced by a provider that only understands SOAP over Java Message Service (JMS).*
- Interface. *Requesters and providers need not agree on a common interface. An ESB reconciles differences by transforming request and response messages into a form expected by the recipient.*
- Qualities of (interaction) service. *Participants or systems administrators declare their quality-of-service requirements, including authorization of requests, encryption and decryption of message contents, automatic auditing of service interactions and how their requests should be routed (optimizing for speed or cost, for example).*

Interposing the ESB between participants enables you to modulate their interactions through a logical construct called a *mediation*. Mediations operate on messages in transit between requesters and providers. For complex interactions, mediations can be chained sequentially.

WebSphere ESB benefits for WebSphere Application Server users

- **Reuse of existing developer skills with an easy-to-use mediation capability**
  WebSphere ESB and WebSphere Integration Developer are designed to provide a rich, integrated, interactive and visual development environment that enables rapid development of integration logic that requires minimal knowledge of Java or J2EE.
- **Simplified administration**
  With WebSphere ESB, true role-based support provides a simplified administration experience. Services-oriented administration requires fewer steps and lessens the need for developer involvement.
- **Faster time to value with prebuilt mediation primitives**
  WebSphere ESB provides prebuilt mediation primitives that are ready to use. Developers simply compose these functions into a mediation flow component to assemble with other components.
- **Extending the reach of existing assets with broad connectivity**
  Along with the variety of connection mechanisms that are found within the ESB, such as SOAP over HTTP, SOAP over JMS and JMS, WebSphere ESB enables developers to connect to existing assets and applications using a client package and extensive WebSphere Adapter support, including JCA adapters. And with WebSphere Integration Developer, the tool that complements WebSphere ESB, users get access to four technology adapters for development and deployment use, as well as development use of four JCA ERP adapters.

ESB in the SOA programming model

IBM has introduced a programming model for implementing services and assembling them into solutions. Service Component Architecture (SCA) and Service Data Objects (SDO) provide the underpinnings for this SOA programming model. SCA defines the model for describing service components and offers a way to assemble them into solutions; SDO defines a model for the information exchanged between these components. SCA and SDO are based on Web services standards such as WSDL and XML Schema Definition (XSD) Language, and augment these interoperability standards to define a component model for SOA. This model is discussed in the IBM developerWorks® article, "Introduction to the IBM SOA programming model," available at **ibm.com**/developerworks/webservices/library/ws-soa-progmodel/.

WebSphere ESB software manages the flow of messages between SCA-described interaction end points and enables the quality of interaction these components request. Mediations within the ESB handle routing and logging functions, as well as mismatches between requesters and providers, including protocol or interaction-style mismatches and interface mismatches. In an overall SCA-based solution, mediations are implemented using a pattern of SCA components that perform a special role, and therefore, have slightly different characteristics than other components that operate at the business level. Mediation components operate on messages exchanged between service end points; in contrast with regular business application components, they are concerned with the flow of the messages through the infrastructure and not just with the business content of the messages. Rather than performing business functions, they perform routing, transformation and logging operations on the messages. The information that governs their behavior is often held in headers flowing with the business messages. The IBM SOA programming model introduces the service-message object (SMO) pattern for SDOs to support this pattern. To learn more about SCA, visit **ibm.com**/ developerworks/library/specification/ws-sca/.

WebSphere ESB and WebSphere Integration Developer

WebSphere ESB provides ESB capabilities in the IBM SOA programming model environment. It facilitates interactions between service end points using different protocols and transports (such as SOAP over HTTP, SOAP over JMS, JMS and J2EE Connector Architecture [JCA] adapters), and it supports mediations between these end points that can transform, log and route messages.

Mediations are created and assembled together with other business components into SCA solutions, using the associated development product, IBM WebSphere Integration Developer. WebSphere Integration Developer helps improve time to value by enabling the use of predefined mediation components within WebSphere ESB, as well as providing adapters to a large set of applications. Together, WebSphere ESB and WebSphere Integration Developer provide four key benefits, which this white paper explores in depth:

- *Web services connectivity, messaging and service-oriented integration*
- *Ease of use throughout the life cycle of a solution, from assembly of ESB-based mediation interactions to testing, deployment and administration of ESB-based solutions*
- *Improved time to value*
- *Seamless integration with products across the WebSphere software portfolio*

Web services connectivity, messaging and service-oriented integration

WebSphere ESB software supports advanced interactions between service end points on three levels: broad connectivity, a spectrum of interaction models and qualities of interaction, and mediation capabilities. The product supports connectivity between end points through a variety of protocols and application programming interfaces (APIs), such as JMS, Version 1.1 implemented in WebSphere platform messaging and IBM WebSphere MQ, SOAP over HTTP Secure (HTTPS) and SOAP over JMS. And with this release, WebSphere ESB, Version 6.0.2 provides significantly improved performance for connectivity.

Because it is built on WebSphere Application Server, WebSphere ESB can provide smooth interoperability with other products in the WebSphere portfolio, including WebSphere MQ and IBM WebSphere Message Broker. And with this release, WebSphere ESB supports native WebSphere MQ binding so that integration with WebSphere MQ and WebSphere Message Broker is much easier and faster than before. WebSphere ESB can also use IBM WebSphere Adapter solutions to take advantage of existing application assets, as well as capture and disseminate business events. With WebSphere Integration Developer, you get access to four technology adapters (e-mail, File Transfer Protocol [FTP], flat file and Java Database Connectivity [JDBC]) for development and deployment use, as well as development use of four JCA enterprise resource planning (ERP) adapters (SAP, PeopleSoft, Oracle E-Business and JD Edwards).

Client interfaces included with WebSphere ESB software further extend this connectivity. The message clients for C/C++ and Microsoft .NET enable non-Java applications to connect to WebSphere ESB using an application programming interface (API) similar to the JMS API. The Web services client for C++ is similar to the Java API for XML-based Remote Procedure Call (JAX-RPC) and enables users to connect to Web services hosted on WebSphere Application Server from within a C++ environment. Other features at the connectivity level perform basic protocol conversion between end points where the protocol used by the requester to dispatch requests (such as SOAP over HTTP) is different from that of the service provider that is to handle those requests (such as SOAP over JMS).

WebSphere ESB supports a range of interaction models, including request-reply, point-to-point and publish-subscribe interactions. It supports Web services standards including WS-Security and WS-Atomic Transactions. It also includes Universal Description, Discovery and Integration (UDDI), Version 3.0 that can be used to publish and manage service end-point metadata, which enables service definitions to be made available to client applications. Integration developers can interact with UDDI to locate and service interfaces when developing mediation modules. WebSphere ESB also supports IBM WebSphere Service Registry and Repository. In particular, the dynamic end-point lookup primitive enables you to query service end-point information and connect to new service providers without redevelopment and redeployment. This integration gives systems the flexibility to consume services along with optimized runtime access that can be governed by approved policies.

WebSphere ESB benefits for WebSphere MQ users

- **Increased flexibility for existing systems**
  WebSphere ESB integrates new environments in an open-standards-based way to seamlessly connect organizations, partners, departments and systems that use Java, Microsoft .NET or Web services standards to enable interoperation with existing WebSphere MQ networks.
- **A single point of control to improve manageability**
  WebSphere ESB combines, in a single package, features to cope with the diversity of today's IT environment by providing Web services-based flexible message routing and transformation options that can be distributed across the network but controlled centrally.
- **Easy-to-use tools**
  WebSphere ESB and WebSphere Integration Developer are tailored to the skills and knowledge of integration specialists to help speed time to deployment.
- **The ability to extend your environment**
  WebSphere ESB provides native support for WebSphere MQ, to complement existing WebSphere MQ environments.

Finally, WebSphere ESB supports mediation of interactions between service end points beyond the protocol transcoding supported by the connectivity functionality. This capability enables integration logic processing to be handled in the ESB rather than in the service end points. WebSphere ESB mediation capabilities also include support for context-based and other forms of routing of messages that are exchanged using an ESB, as well as other operations, such as logging or transformation of messages.

Ease of use

WebSphere ESB and WebSphere Integration Developer are designed to get users up and running quickly with comprehensive documentation, easy-to-understand samples and a compelling experience. WebSphere Integration Developer provides an easy-to-use tool that requires minimal programming skills to perform the typical tasks of an integration developer when designing, testing, configuring and deploying ESB-based solutions. Integration developers use graphical tools to identify and connect service end points, and optionally build mediation flows on those connections. Mediation flows are visually composed from mediation primitives that are chosen from a palette, configured and connected together. These primitives include support for message routing, enrichment, logging and transformation. Integration developers can then locally test and debug mediated interactions in the WebSphere Integration Developer environment before deploying them in the WebSphere ESB runtime environment (see Figure 1). At run time, the WebSphere ESB administration console enables solution administrators to manage WebSphere ESB deployments with new role-based administration support that provides a simplified user experience.

Reduced time to value

WebSphere ESB provides an easy-to-use, cost-effective solution for services integration that helps users to exploit existing IT investments by quickly building a flexible integration infrastructure to extend the value of these investments. The product's extensive business and IT standards facilitate greater interoperability and portability, enabling users to take advantage of first-class support for hundreds of independent software vendor (ISV) solutions through extensive WebSphere Adapter support, including new JCA-based adapters.

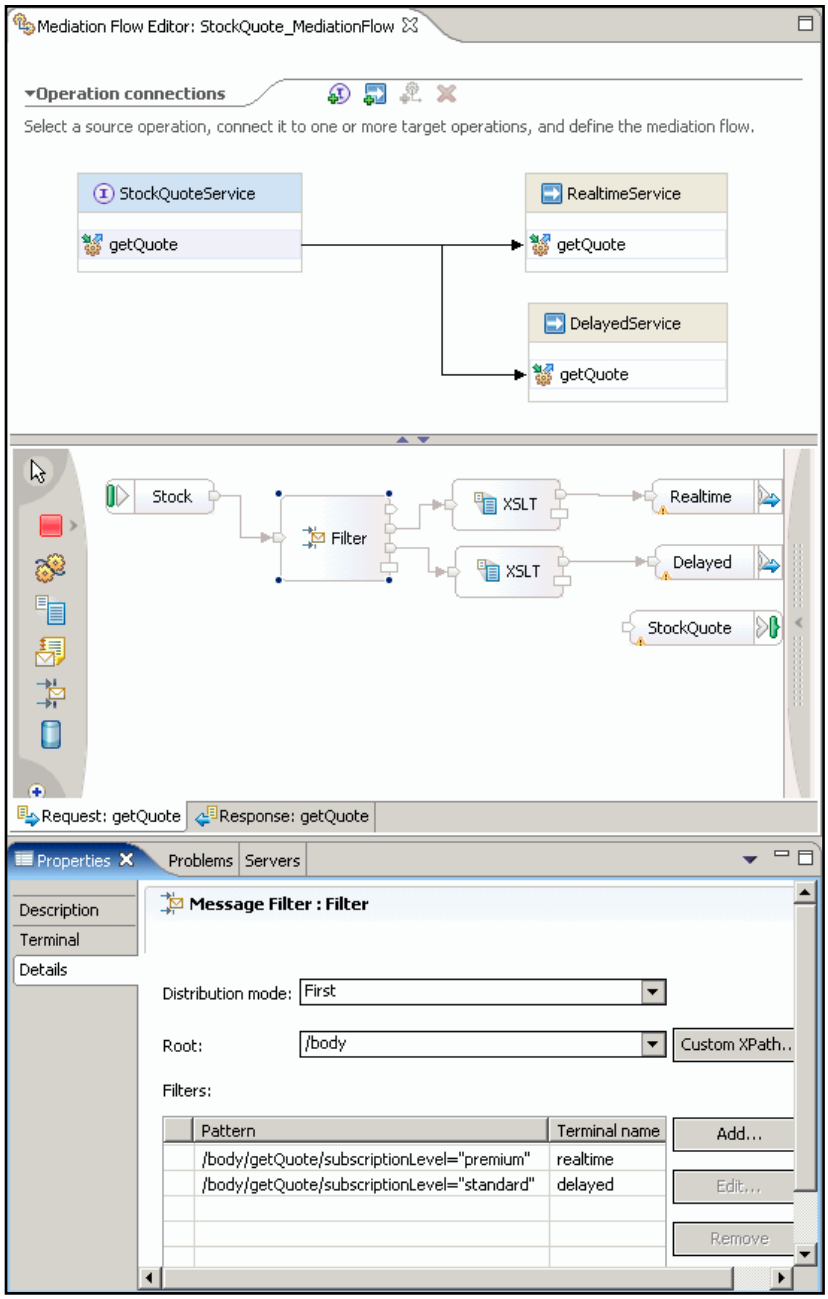

Figure 1. WebSphere Integration Developer mediation flow editor

WebSphere ESB managed interactions can also be efficiently reconfigured to meet changing business-processing loads. And the integration developer or system administrator can dynamically add or replace end points without affecting the rest of the ESB-based solutions. You can also feed selected events for monitoring with new common-event infrastructure (CEI) support and integration with WebSphere Business Monitor. The WebSphere ESB administration console provides full access to the underlying WebSphere Application Server administration capabilities through progressive disclosure of functions.

Seamless integration with other products across the WebSphere platform

WebSphere ESB takes full advantage of the capabilities of the underlying IBM WebSphere Application Server Network Deployment platform and inherits that product's qualities of service, workload-balancing, clustering, high-availability and scalability features. Through this deep integration with WebSphere Application Server, WebSphere ESB also inherits integration with IBM Tivoli® security, directory and systems-management offerings, and includes IBM Tivoli Access Manager (for optional use, to deliver a highly secure, unified and personalized experience) and IBM Tivoli Directory (for optional use, as a Lightweight Directory Access Protocol [LDAP] server). WebSphere ESB also integrates with IBM Tivoli Composite Application Manager for SOA, providing monitoring of Web services messages and management of their end points. Tivoli Composite Application Manager for SOA feeds service-performance information to WebSphere Service Registry and Repository, giving WebSphere ESB better control for its dynamic selection of service end points.

By sharing the same administrative console as WebSphere Application Server and WebSphere Process Server, WebSphere ESB helps extend a familiar interface across the operational control of the family of products, and enables a single administrator to manage them all. Also, WebSphere ESB can be combined with an existing WebSphere MQ messaging installation to integrate existing messaging backbones into new environments using open standards. And because WebSphere ESB interoperates with WebSphere Message Broker, you can implement complex ESB topologies with WebSphere ESB handling standards-based Web service interactions and WebSphere Message Broker providing its advanced support for a wide range of message formats.

You can upgrade from WebSphere ESB to WebSphere Process Server as your ESB requirements increase, adding support for advanced integration capabilities such as business processes and state machines for end-point orchestration, and business rules for dynamic decision making. The integration developer uses WebSphere Integration Developer for both run times, so the development environment scales with your needs. Similarly, the administrative console scales from WebSphere Application Server to WebSphere ESB and WebSphere Process Server.

WebSphere ESB benefits for WebSphere Message Broker users

- **Improved performance and optimized networks**
  WebSphere ESB enables message-processing capabilities to be efficiently (and cost-effectively) deployed into branches, terminals, warehouses, stores and so on.
- **Maximized business flexibility**
  WebSphere ESB includes a decoupled approach that enables stores and other entities to run independently to improve business flexibility.
- **Enabling asset reuse**
  WebSphere ESB supports Web services standards.
- **Use of existing assets**
  WebSphere ESB complements existing WebSphere Message Broker investments, and integrates with WebSphere MQ native protocols and formats.
- **Reuse existing developer skills**
  Because the WebSphere ESB and WebSphere Message Broker mediation-development models are very similar, integration developers do not have to learn new skills.

Creating and administering an ESB solution with WebSphere Integration Developer

One of the best ways to appreciate the mediation capabilities of WebSphere ESB is by understanding how the product enables users to create and manage an ESB solution.

ESB user roles and their tasks

IBM introduces two user roles that create and manage ESB-based solutions:

- *An* integration developer *uses ESB-related tools and technology to define end points, and connect and build the logic that controls how requests are routed between services. The individuals in these roles understand the semantics of the business services to be integrated, and focus on creating the mediation modules that enable the interactions. The integration developer uses WebSphere Integration Developer to fulfill his or her role.*
- *A* solution administrator *makes new SOA solutions available by deploying any new services required, as well as the mediations that enable new and existing services to interact correctly. This person understands the basic interaction patterns of an organization's business processes, and the behaviors required from the overall solution. The solution administrator can adjust the configuration of a deployed solution, reacting to the observed behavior as monitored by the operators of the IT systems. The solution administrator uses the capabilities provided by the administration console in WebSphere ESB, and the underlying and embedded capabilities of the administration console for WebSphere Application Server.*

Creating the mediation function required to allow the end points to interact effectively

The integration developer uses WebSphere Integration Developer to create a mediation module, which specifies the end points that are to be integrated, and the bindings, or connectivity protocols, that are to be used to connect to it. The processing required for a message as it flows through the mediation module is defined by selecting and assembling mediation primitives. WebSphere Integration Developer provides a palette of predefined mediation primitives, which include:

- Fail, *which throws an exception and ends the path through the mediation flow.*
- Stop, *which silently ends the path through the mediation flow.*
- MessageElementSetter, *which provides an easy way to change data in a mediation flow without custom coding.*
- MessageFilter, *which compares the content of the message to a list of XPath expressions configured by the user, and routes the message to the next mediation primitive based on the result.*
- ServiceRegistryLookup, *which selects an end point from the WebSphere Service Registry and Repository to connect to.*
- Extensible Stylesheet Language Transformations (XSLT), *which transforms messages according to transformations defined by an XSL style sheet.*
- DatabaseLookup, *which searches values from a database and stores them as elements, identified by XPath expressions, in the message.*
- MessageLogger, *which logs an XML copy of the message to a database for future retrieval or audit.*
- EventEmitter, *which generates a Common Base Event from a mediation (that can be consumed by WebSphere Business Monitor).*

The integration developer customizes the message primitives, for example by naming the database to be searched or providing the XSL style sheet.

Configuring these mediations does not require programming. Because WebSphere Integration Developer is designed to hide the complexity of WSDL, XML Schema, XPath and XSLT, integration developers need not be skilled in these core ESB technologies to build fully capable SOA solutions. If the supplied primitives do not meet your needs, advanced users can use the custom mediation primitive to author Java code, either directly or visually, within an SCA Java component. The full SCA and SDO programming model and API are available, as well as specific System Programming Interface (SPI) functionality for mediation authors.

If the interaction follows the request-response paradigm, whether synchronous or asynchronous, the integration developer can create a response mediation flow, using the same WebSphere Integration Developer capabilities as those used to create the request mediation flow. Finally, a visual debugger enables the integration developer to debug both request and response mediation flows, with breakpoints, single-stepping and inspection of the fields of messages as they flow through the mediation.

Deploying, administering and managing mediation modules

Mediation modules are deployed to WebSphere ESB using service deploy, the WebSphere ESB deployment tool. Because WebSphere ESB administration is based on the WebSphere Application Server administration console, it has all the capabilities necessary to manage the ESB, available in one familiar interface, integrated with the underlying application server. However, the solution administrator can choose to apply the application integration task filter, which can restrict the task list to those tasks that are relevant to his or her role. At any time, the solution administrator can choose to expose the full function of the WebSphere Application Server administration console again.

Mediation modules share instrumentation with other WebSphere Application Server artifacts, and their individual performances can be monitored through IBM Tivoli Performance Monitor, which is available as part of WebSphere ESB. Also, services and mediations connected through Web services bindings can be monitored using IBM Tivoli Composite Application Manager for SOA, which can track messages as they flow through the ESB, and to and from other Web services, and can monitor message rates and response times, raising alerts when they do not meet target values.

Conclusion

SOA is the next evolutionary step in IT architecture developed to help organizations meet their increasingly complex challenges, by taking advantage of its existing investment in developers, software languages, hardware platforms, databases and applications to help reduce costs and risks while boosting productivity. This adaptable, flexible architecture provides the foundation for shorter time to market, and helps reduce costs and risks in development and maintenance.

One lesson learned, time and time again over the past three decades (or more), is that old applications do not disappear quickly. They persist because they work. As a result, the need for the flexibility of an ESB in performing the transformations, routing and interconnection between existing and new applications is critical. WebSphere ESB provides ESB capabilities in the SOA programming-model environment. It facilitates interactions between service end points using a wide range of protocols, and supports mediations between these end points that can transform, log and route messages.

WebSphere Integration Developer, the development tool for WebSphere ESB, provides an integrated, interactive and visual development experience that requires minimal programming skills. Integration developers can get up and running quickly with comprehensive documentation, easy-to-understand samples and a compelling experience. Development is made easier by simplifying the ability to declare services and define their interconnections, and by the ability to visually compose mediation functions with first-class tool support for intelligent message routing, enrichment and transformation. And role-based administration support makes it easy to manage WebSphere ESB deployments through a simplified user experience for solution administrators.

WebSphere ESB software can also help improve your time to value. As a cost-effective solution for services integration, WebSphere ESB helps you use your SOA IT investments by building a flexible integration infrastructure that has the potential to extend the value of your existing investments, regardless of vendor. Support for hundreds of ISV solutions and extensive WebSphere Adapter support make it easy to connect with your existing assets. Prebuilt mediation function helps reduce development time and costs. And because WebSphere ESB is based on WebSphere Application Server, you can take advantage of market-leading qualities of service, such as clustering, failover, systems management and security. Common tooling and administration means the move from WebSphere ESB to WebSphere Process Server is virtually seamless. And integration with Tivoli software provides world-class security and systems-management capabilities. By providing these four key areas of value, WebSphere ESB software helps make SOA a reality for your business.

For more information

To learn more about the SOA programming model, visit:

ibm.com/developerworks/library/ws-soa-progmodel4/

To learn more about IBM WebSphere ESB, contact your IBM representative or IBM Business Partner, or visit:

ibm.com/software/integration/wsesb/

To learn more about IBM WebSphere Integration Developer, contact your IBM representative or IBM Business Partner, or visit:

ibm.com/software/integration/wid

To join the Global WebSphere Community, visit:

www.websphere.org

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Produced in the United States of America
11-06

WSW11278-USEN-00